DÉCODAGE DES CODES DE REED-MULLER D'ORDRE 1

par

Robert Rolland

Résumé. — On décrit ici le décodage des codes de Reed-Muller d'ordre 1, et notamment la version en temps linéaire dûe à S.N. Litsyn et O.I. Shekhovtsov dans [**1**].

Table des matières

1. Introduction

On rappelle brièvement la définition suivante des codes de Reed-Muller d'ordre 1. On note $\mathbb{F}_2$ le corps à deux éléments $\{0,1\}$. Soit $m \geq 2$ un entier. On note $\mathbf{F}$ l'espace de toutes les fonctions de $\mathbb{F}_2^m$ dans $\mathbb{F}_2$. On munit cet espace de la base des évaluations $(e_z)_{z\in\mathbb{F}_2^m}$ où

$$e_z(x) = \begin{cases} 1 & \text{si} \quad x = z \\ 0 & \text{si} \quad x \neq z \end{cases} .$$

Dans cette base toute fonction f s'écrit sous la forme

$$f = \sum_{x\in\mathbb{F}_2^m} f(x)e_x.$$

Autrement dit on décrit f par ses images. Le code de Reed-Muller d'ordre 1 correspondant est le sous-espace $\mathbf{RM}(1,m)$ de $\mathbf{F}$ constitué par les fonctions affines. Une fonction affine s'écrit sous la forme

$$f(x) = u_0 + u_1x_1 + \cdots u_mx_m. \tag{1}$$

Le sous-espace $\mathbf{RM}(1,m)$ est donc un sous-espace de dimension $k = m+1$. On peut utiliser ce code de la façon suivante : on part d'un mot brut $(u_0, u_1, \cdots, u_m)$ de longueur $k = m+1$, à ce mot correspond la fonction affine $f \in \mathbf{RM}(1,m)$ définie par (1). Le mot encodé est alors donné par la décomposition de f dans la base $(e_z)_z$ c'est-à-dire le mot $(f(x))_{x\in\mathbb{F}_2^m}$.

2. Les paramètres

Le code $\mathbf{RM}(1,m)$ a donc pour dimension $k = m+1$, pour longueur $n = 2^m$. On peut décrire facilement le poids des mots du code. En effet, si la fontion affine f est nulle, le poids du mot de code correspondant est 0. Si la la fontion affine f est 1, le poids du mot correspondant est 2^m. Dans tous les autres cas, l'ensemble des zéros de f est un hyperplan affine qui possède donc 2^{m-1} points, et le poids du mot correspondant est donc $2^m - 2^{m-1} = 2^{m-1}$. La distance minimale du code est donc $d = 2^{m-1}$. On en conclut que ce code peut corriger t erreurs avec

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor = \left\lfloor 2^{m-2} - \frac{1}{2} \right\rfloor = 2^{m-2} - 1.$$

3. Transformée d'Hadamard et décodage

Si ϕ est une fonction de $\mathbb{F}_2^m$ dans $\mathbb{C}$, on définira la transformée d'Hadamard de ϕ par

$$\hat{\phi}(u) = \sum_{x\in\mathbb{F}_2^m} \phi(x)(-1)^{\langle u,x\rangle} \tag{2}$$

où $\langle u,x\rangle = \sum_{i=1}^{2^m} u_ix_i$. On trouvera une étude détaillée de cette transformation dans [**2**]. Comme toujours dans ces questions on peut mettre ou ne pas mettre de constante multiplicative devant la somme pour des questions d'orthonormalisation. Ici il vaut mieux mettre le coefficient 1.

Soit f un mot de code. On peut écrire f sous la forme

$$f(x) = u_0 + \langle u, x\rangle.$$

Associons à ce mot la fonction

$$F(x) = (-1)^{f(x)} = (-1)^{u_0}(-1)^{\langle u,x\rangle}.$$

On voit alors qu'au signe près $F(x)$ est un caractère du groupe $\mathbb{F}_2^m$ et donc ses produits scalaires avec les autres caractères sont nuls sauf avec lui-même. En conséquence tous les coefficients d'Hadamard sont nuls sauf celui d'indice u.

On a donc le résultat suivant

$$\hat{F}(v) = \begin{cases} (-1)^{u_0}2^m & \text{si} \quad v = u \\ 0 & \text{si} \quad v \neq u \end{cases} \tag{3}$$

Pour ne pas surcharger les calculs supposons que $u_0 = 0$. On suppose que dans une transmission, au lieu du mot de code f on reçoit le mot erroné g. On suppose que le nombre d'erreurs est $\leq t$. Ainsi

$$g(x) = f(x) + e(x).$$

Introduisons le support S de $e(x)$

$$S = \{x \mid e(x) = 1\}.$$

Par hypothèse sur le nombre d'erreurs, on a

$$\epsilon = \#S \leq t.$$

Notons $G(x) = (-1)^{g(x)}$ et calculons la transformée d'Hadamard de $G(x)$.

$$\hat{G}(v) = \sum_{x\in\mathbb{F}_2^m} (-1)^{\langle u,x\rangle+\langle v,x\rangle+e(x)},$$

$$\hat{G}(v) = \sum_{x\in\mathbb{F}_2^m} (-1)^{\langle u+v,x\rangle+e(x)},$$

$$\hat{G}(v) = \sum_{x\notin S} (-1)^{\langle u+v,x\rangle} - \sum_{x\in S} (-1)^{\langle u+v,x\rangle},$$

$$\hat{G}(v) = \sum_{x\in\mathbb{F}_2^m} (-1)^{\langle u+v,x\rangle} - 2\sum_{x\in S} (-1)^{\langle u+v,x\rangle}.$$

(1) Si $u = v$ on a

$$\hat{G}(u) = 2^m - 2\epsilon.$$

(2) Si $u \neq v$ on a

$$\hat{G}(v) = -2\sum_{x\in S}(-1)^{\langle u+v,x\rangle},$$

et donc

$$\left|\hat{G}(v)\right| \leq 2\epsilon \leq 2t = 2^{m-1} - 2.$$

Étudions la différence $\hat{G}(u) - \left|\hat{G}(v)\right|$ lorsque $v \neq u$.

$$\hat{G}(u) - \left|\hat{G}(v)\right| \geq 2^m - 4\epsilon,$$

$$\hat{G}(u) - \left|\hat{G}(v)\right| \geq 2^m - 4t = 4.$$

En conclusion on obtient la relation suivante

$$\hat{G}(u) \geq \max_{v\neq u}\left|\hat{G}(v)\right| + 4.$$

Si maintenant $u_0 = 1$ on obtient une relation analogue au signe près, si bien que dans tous les cas on peut écrire

$$\left|\hat{G}(u)\right| \geq \max_{v\neq u}\left|\hat{G}(v)\right| + 4. \tag{4}$$

La relation (4) fournit une stratégie de décodage

(1) Ayant reçu le mot g on calcule $G = (-1)^g$.

(2) On calcule la transformée d'Hadamard de G.

(3) On repère le $u \in \mathbb{F}_2^m$ tel que $\left|\hat{G}(u)\right|$ soit maximal. Ceci donne les coefficients $u_1, \cdots u_m$ de la fonction f.

(4) Si $\hat{G}(u) > 0$ alors $u_0 = 0$, sinon $u_0 = 1$.

4. Pratique du décodage

On peut calculer la transformée d'Hadamard par l'algorithme de transformation d'Hadamard rapide (FHT) qu'on trouvera détaillé dans [**2**] et dont on redonne ici le dessin du papillon.

Mais comme dans notre cas on veut seulement trouver le coefficient de plus grande valeur absolue, nous allons voir qu'il n'est pas nécessaire de calculer toute la transformation.

Pour expliquer ce qu'il se passe, reprenons l'algorithme de la transformation et introduisons les notations suivantes : Soit T_0 le tableau initial

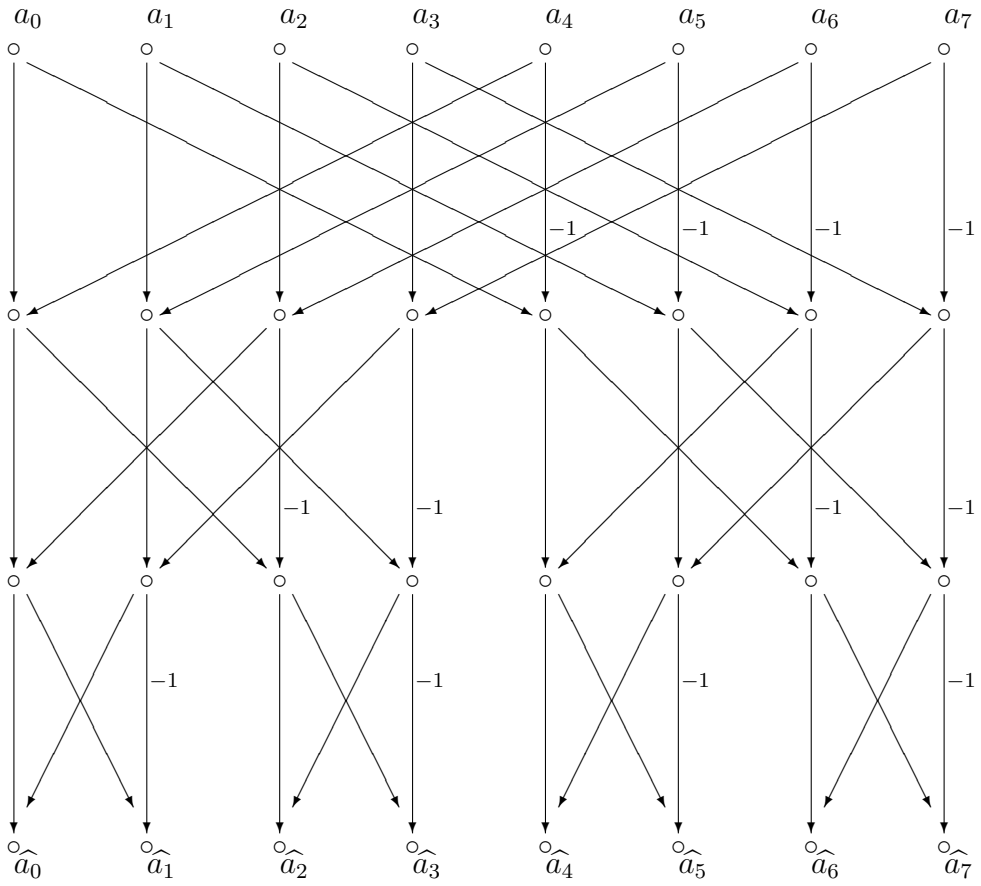

FIGURE 1. La FHT sur 8 points

de taille 2^m dont on veut faire la transformation d'Hadamard. Ce tableau contient les éléments $a_0, a_1, \cdots, a_{2^m-1}$. Au premier pas de la transformation on obtient un tableau T_1 constitué de deux sous-tableaux $T_{1,1}, T_{2,1}$ qui sont la première moitié et la deuxième moitié du tableau T_1. Le tableau $T_{1,1}$ contient les $a_i + a_{i+2^{m-1}}$ tandis que le tableau $T_{2,1}$ contient les $a_i - a_{i+2^{m-1}}$ $(0 \leq i \leq 2^{m-1} - 1)$. On itère ce procédé, ce qui consiste au deuxième pas à introduire les tableaux $T_{1,2}, T_{2,2}, T_{3,2}, T_{4,2}$ et ainsi de suite jusqu'au dernier pas (le pas m) où on obtient 2^m tableaux de taille 1, chacun contenant un coefficient d'Hadamard.

Si f est un mot de code, et si $F = (-1)^f$, à chaque étape tous les tableaux sauf un ne contiennent que des zéros. Plus précisément, si à

l'étape j, le tableau qui ne contient pas que des zéros est $T_{i,j}$, à l'étape $j+1$ le tableau qui ne contient pas que des zéros est l'un des deux tableaux $T_{2*i-1,j+1}$ ou $T_{2*i,j+1}$. Le poids, c'est-à-dire la somme des valeurs absolues des coefficients, des tableaux « non nuls » est toujours 2^m.

Soit maintenant g le mot f perturbé par au plus t erreurs. Remarquons que chaque élément d'un tableau est une combinaison linéaire des valeurs de départ a_i et qu'on peut mettre cet élément sous la forme

$$\alpha = \sum_{i \in S_\alpha} \pm a_i.$$

De plus dans un même tableau deux éléments α et β ont des supports S_α et S_β disjoints. En conséquence chaque tableau construit à partir de g diffère du tableau correspondant construit à partir de f d'un poids au plus $2t$. Nous avons déjà vu que $2^m - 2t > 2t$, en conséquence, les tableaux qui étaient de poids maximum pour f restent les tableaux qui à chaque étape ont le poids maximum pour g. Ceci nous permet à chaque étape de la transformation de ne garder que le tableau de poids maximum et d'écarter les autres. Ainsi à la première étape on fait une transformation sur 2^m éléments, à la deuxième une transformation sur 2^{m-1} éléments et à l'étape m on termine par une transformation sur 2 élément. On peut vérifier facilement que l'algorithme est en temps linéaire par rapport à 2^m. En effet à chaque étape j on fait 2^{m-j} additions, 2^{m-j} soustractions, 2 calculs de poids et une comparaison afin de déterminer le tableau à choisir pour l'étape suivante. Tout ceci donne une dépendance linéaire en 2^m du fait que $2^m + 2^{m-1} + \cdots + 1 < 2.2^m$.

Prenons un exemple avec $m = 4$. Soit $f(x) = x_1 + x_2 + x_4$. On calcule les valeurs de f

$$\begin{array}{cccc}
f(0,0,0,0) = 0 & f(0,0,1,0) = 0 & f(0,0,0,1) = 1 & f(0,0,1,1) = 1 \\
f(1,0,0,0) = 1 & f(1,0,1,0) = 1 & f(1,0,0,1) = 0 & f(1,0,1,1) = 0 \\
f(0,1,0,0) = 1 & f(0,1,1,0) = 1 & f(0,1,0,1) = 0 & f(0,1,1,1) = 0 \\
f(1,1,0,0) = 0 & f(1,1,1,0) = 1 & f(1,1,0,1) = 1 & f(1,1,1,1) = 0
\end{array}$$

Regardons la transformation d'Hadamard sur f restreinte aux tableaux de plus grand poids

$$\begin{array}{cccccccccccccccc}
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & -2 & 2 & 2 & -2 & -2 & 2 \\
 & & & & & & & & 4 & -4 & -4 & 4 & 0 & 0 & 0 & 0 \\
 & & & & & & & & 0 & 0 & 8 & -8 & & & & \\
 & & & & & & & & & & 0 & 16 & & & &
\end{array}$$

On va maintenant refaire le même calcul avec la fonction G obtenue à partir de F en modifiant $t = 3$ valeurs a_2, a_{10}, a_{14}.

$$\begin{array}{cccccccccccccccc}
 & & \downarrow & & & & & & & & \downarrow & & & & \downarrow & \\
1 & -1 & 1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 2 & -2 & 2 & 2 & 2 & -2 & 0 & 2 \\
 & & & & & & & & 4 & -4 & 2 & 4 & 0 & 0 & 2 & 0 \\
 & & & & & & & & 6 & 0 & 2 & -8 & & & & \\
 & & & & & & & & & & -6 & 10 & & & &
\end{array}$$

On décode donc en prenant $u = (u_1, u_2, u_3, u_4)$ tel que $\left|\hat{G}(u)\right|$ soit maximum. Ici ce u représente l'indice 11 (il est à la douzième place, mais ça part de zéro), c'est-à-dire $u = (1, 1, 0, 1)$. De plus, $u_0 = 0$ car le coefficient qui atteint le maximum en valeur absolue est > 0 (ce coefficient vaut 10). La fonction

$$f(x) = u_0 + u_1x_1 + u_2x_2 + u_3x_3 + u_4x_4 = x_1 + x_2 + x_4$$

est donc bien retrouvée au décodage.

Références

[1] S. Litsyn & O. Shekhovtsov – « Fast decoding algorithm for first-order Reed-Muller codes », *Problemy Peredachi Informatsii* **19**, no. 2, p. 3–7.

[2] R. Rolland – « Remarques sur le calcul de la transformation d'Hadamard », *Dossiers Acrypta* (2008).

---

31 Décembre 2008

R. Rolland, Association ACrypTA, 50 Rue Edmond Rostand 13006 Marseille,
E-mail : robert.rolland@acrypta.fr